Instrumenting the Crowd: Using Implicit Behavioral Measures to Predict Task Performance

Jeffrey M. Rzeszotarski, Aniket Kittur
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA
{jeffrz, nkittur}@cs.cmu.edu

ABSTRACT
Detecting and correcting low quality submissions in crowdsourcing tasks is an important challenge. Prior work has primarily focused on worker outcomes or reputation, using approaches such as agreement across workers or with a gold standard to evaluate quality. We propose an alternative and complementary technique that focuses on the way workers work rather than the products they produce. Our technique captures behavioral traces from online crowd workers and uses them to predict outcome measures such quality, errors, and the likelihood of cheating. We evaluate the effectiveness of the approach across three contexts including classification, generation, and comprehension tasks. The results indicate that we can build predictive models of task performance based on behavioral traces alone, and that these models generalize to related tasks. Finally, we discuss limitations and extensions of the approach.

ACM Classification: H5.2 Information interfaces and presentation (e.g., HCI) --- User Interfaces: Evaluation / methodology, Theory Methods;

General terms: Human Factors, Measurement

Keywords: Crowdsourcing, Mechanical Turk, Event Logging, User Logging, User Behavior, Performance

INTRODUCTION

Crowdsourcing markets like Amazon's Mechanical Turk (MTurk) allow users to rapidly disseminate large quantities of small tasks to a large pool of willing workers. This empowers researchers to assemble large datasets of human labeled corpora, corporations to outsource simple data processing, and even one day to have individuals utilize crowdworkers to complete tasks in their own word processors [1, 2]. The ability to quickly and effectively reach a willing microtask work force has the potential to change the way work is done in society.

However, the distributed nature of such markets can pose challenges for employers. Because tasks are typically small, short, and high volume, workers can expend minimal effort or even cheat on jobs as their output often blends in with the crowd. This is especially true for subjective tasks or those with multiple valid answers, which can attract cheating rates of over 30% [15]. Adding to this issue is the limited reputation system in MTurk which only tracks the total percentage of work a worker has had accepted; cheaters can slip through and even maintain high reputations by accepting tasks they are unlikely to get rejected for. Even if workers are not cheating, there can be high variability in the quality of their work due to differences in effort or skill [2].

Significant research efforts have been made to develop ways to detect and correct for low quality work in order to improve the overall quality of the resulting data. Researchers have proposed a variety of approaches to address this issue, ranging from using gold standards to post-hoc weighting based on worker agreement or reputation [2, 4, 12]. Most of these approaches rely on a single aspect of the workflow in human computation markets: the end products. With only the end products of the work process and some minimal reputation metrics about the workers involved, employers must make difficult tradeoffs depending on the quality control method they use. For example, methods based on worker agreement rely on multiple redundant worker judgments, while gold standards require some percentage of labeled data.

We present an alternative and complementary technique for evaluating task performance on crowdsourcing markets: examining the way the workers work, rather than the products they produce. The work process itself may provide information as to the effort, skills, and behavior of the worker which may be useful for predicting their output quality. For example, imagine two different workers tasked with tagging an image. The lazy worker, after accepting the task, may immediately scroll to the text fields, type their first tag, tab to the second field, and without delay enter their second tag. The conscientious worker, after accepting the task, may pause while inspecting the image, scroll to the text fields, type their first tag, scroll back to the image, pause again, and then type their second tag. An employer presented only with the behavioral patterns of each worker might be able to tell the conscientious worker from the lazy worker, and weight their judgments accordingly.

In this paper we examine whether crowdworkers' behavioral traces as they complete work can actually be used to predict the quality of their final product. We propose a technique we call *task fingerprinting* to collect and analyze such behavioral traces in online task markets (though the technique may generalize to other settings as well). We describe a prototype implementation of task fingerprinting on the Amazon Mechanical Turk market, and test its efficacy in three different task contexts. Our results demonstrate how workers' behavioral traces can be used to make inferences about their task performance, including identifying cheaters, estimating output quality, and predicting errors. Finally, we discuss some limitations and extensions of the task fingerprinting approach, and explore applications beyond the crowdsourcing context.

TASK FINGERPRINTING

A task involves a worker performing some actions on an input (typically provided by the employer) resulting in some output. The input might be an image to tag, a document to summarize, or even just a set of guidelines for open response. Using this input, the worker engages in a series of cognitive and motor actions that result in changes in their web browser (e.g., mouse movements, scrolling, keystrokes) and produces an end product for the requester. This process can be represented as:

$$f_{worker}(\text{input}_{task}) = \text{output}_{task,\ worker}$$

where the input is given by the employer, some sequence of cognitive and motor actions are performed by the worker (f_{worker}) on the input, generating some output that is consumed by the employer. Common methods for quality control alter the design of the input or evaluate the output side of the function, since the cognitive effort and skills of the worker (in f) are not directly observable. Yet, if evaluation based on the *process* of generating results were effective, it would result in a number of benefits. In contrast to gold standard approaches, one could make inferences about the quality of the output even without labeled data; or even having to inspect the output at all. And unlike output agreement approaches, predictions of quality could be made without many redundant judgments from different workers. Furthermore, assuming workers are consistent in their behaviors across tasks (which we examine in more detail later), we could use information about their work process on one task to make inferences about their work on other tasks. For example, we could identify workers that ignore the guidelines of one task so that we could flag all of their work across all tasks for closer examination.

We propose *task fingerprinting* as a means to capture the process that crowdworkers use to complete a task, as embodied by their interactions with their computer interface. By logging what workers do in their interface while working on the task, we can develop a quantitative description of their process which will allow us to compare workers and evaluate end products, make inferences about worker cognition, and determine how effective a worker pool is at a given task and set of data.

Task fingerprints can assume a variety of structures to quantitatively describe what workers do. In their raw form, they are sequential logs of interface events; what the workers did, and when. The sequences encode valuable information, such as the order of operations, time delays between actions, and patterns of labor. Refining this raw data, we can gather summary statistical data, such as counts of different actions or the occurrence of outlier behaviors like copy-pasting, that can be used to compare workers. Machine learning based on the input and fingerprint can be used to infer characteristics of the output such as its probable quality or the likelihood that the worker was cheating. Visualization of the fingerprints might enable human outlier and pattern detection in large sets of workers.

RELATED WORK

Our work is related both to evaluation efforts in crowdsourcing and to user event logging.

Evaluating Crowdsourcing Outcomes

There are at least two general approaches researchers have explored for obtaining good data from crowdworkers. Pre-task approaches focus on designing tasks so that they are resistant to poor responses (e.g., von Ahn and Dabbish). For example, in the context of MTurk, Kittur et al. propose that tasks be designed in such a way that performing poorly or cheating is as costly as contributing in good-faith [15]. Other approaches include promoting intrinsical motivation [18], splitting larger tasks into small, fault-tolerant subtasks [1, 19], incorporating randomness in cooperative task designs [23], financial manipulation and tweaking outcome measures [10, 16]. While these can be effective strategies, they require that tasks be specially tailored for the approach.

Other researchers address the low quality data problem using a post-hoc approach, controlling or correcting for workers' outputs. One common way is to use validated gold standard data to sort out good workers from bad [2, 5]. However gold standard data is not always available or applicable, as is the case in some generative tasks like tagging and summarization. Other work calculates relationships between worker answers or identifies erroneous workers using trends to reduce bias [2, 4, 12, 20]. A congruent approach is to have workers rate other workers in a verification step [1]. While these techniques can be effective, they also have drawbacks; for example, approaches based on worker agreement require a number of redundant judgments, may be susceptible to gaming or majority effects, and may not work well when there are a broad range of valid answers or most answers are unique.

Our methodology of harnessing workers' implicit behaviors provides a number of advantages over the above approaches. First, models of user behavior can generalize across tasks. Second, collecting additional data about the worker's behavior has the potential to improve predictions beyond the theoretical limits of just using a worker's identity and their end products. Third, our method doesn't require knowledge of 'correct' answers, and supports having

Diligent	S_{191px}	M_{33x52y}	C_{Tag1}	$D_{1.1s}$	$K_{OUTSIDE}$	$S_{=198px}$	$D_{1.8s}$	S_{89px}	C_{Tag2}	$K_{PLAYING}$	S_{40px}	C_{Submit}
Lazy	S_{185px}	M_{44x51y}	C_{Tag1}	$K_{MAN<tab>}$	K_{DOG}	S_{50px}	$D_{5.3s}$					C_{Submit}

S=Scroll$_{Amount}$ M=Mouse Movement$_{Distance}$ D=Delay$_{TimePassed}$ C=Click$_{Target}$ K=Keypress$_{Key}$

Figure 1: Example refined event logs for tagging an image with both 'lazy' and 'diligent' workers. The lazy worker quickly writes simplistic tags, while the diligent worker takes time to think and check the source image between tags.

a range of valid answers. Fourth, it can scale down to a small worker pool, making judgments even about individual workers.

On the other hand, our technique falls victim to some of the same problems that affect post-hoc analysis, including majority effects and automated task completers. The method can be improved by combining it with other techniques (e.g. including gold standard data), and we will discuss additional ways to improve our technique in the limitations section.

User Event Logging

HCI researchers have spent considerable effort on capturing and analyzing event logs, primarily for the purposes of evaluating usability [9]. Kim et al. used event logs from video games to remove player frustrations by identifying game components that were causing trouble [14]. Chi et al. used simulated logs to visualize the usability of web pages [3]. Ivory and Hearst evaluated a large number of automation systems, concluding that although automated logging is still not widely implemented, it has cost benefits and allows for quick design comparisons [13].

Event logs have also been used to predict cognitive and user outcomes, for example detecting gender differences in problem solving [6], survey logging [21], predicting age and cognitive impairment [11], and skill levels of users for adaptive interfaces, and estimating the complexity of tasks [7, 17].

Other research applies event logs to process mining. Rather than log small granular actions, workflow and process mining examines logs of larger transactions. These systems allow users to model behavioral patterns among workers and determine worker processes necessary to complete certain tasks [22].

The previous findings indicate that not only do user event logs encode information about performance, but also about skill and user behavior. Our work extends this research to examine the feasibility of using user event logs in a crowdsourcing context across different types of tasks, with a focus not on the usability of the system but instead on predicting the quality of task outcomes.

IMPLEMENTATION

We prototyped a task fingerprinting system that uses Javascript and the jQuery library to monitor user activity on crowdsourcing market web pages. Each time the user clicks within the page, presses a key, scrolls, changes focus, or moves their mouse, an event is triggered and recorded to a list along with a unique user hash, a page hash, event information such as mouse position or which key pressed, and a timestamp (to the millisecond). After completing the task, the user uploads the collected log to a server through an opt-in button. The server uses the Django web framework to record each event in the usage data as a row in an SQLite database for later analysis. The system is portable and able to log users on any website, however the database must be hosted by the web site, as cross site scripting limitations make uploading log data very difficult otherwise.

Event logs are discretized on the server to facilitate analysis, with sequences of scrolling and mouse movement encoded into individual events for each 200 pixels total moved or scrolled. We found it useful to make two refinements to the above approach. First, repeated sequential events, such as mouse movements or scrolling, are encoded into individual events with aggregate information (total mouse movement from start to end, total scrolled position); this avoids simple "spoofing" attacks such as extended scrolling or mouse movement without other activity (Figure 1). Second, discretization may miss significant delay information (for example, if the user scrolls, then reads without moving their mouse, then scrolls again). To address this, we use *delay* events to encode temporal information into the log: if a user waits longer than a specific time threshold (here we use 200 milliseconds) a delay event is encoded, with further delay events added for every 200 milliseconds the user waits.

In addition, we also collect information that characterizes the user's behavior in a holistic sense (see Table 1). Firstly, we generate summary data, such as the total time the system was logging activity, the counts of different types of events, the total amount of scrolling and mouse movement, and the lengths of the raw and collapsed event logs. These allow us to get a general sense of what a user is doing in the environment. Secondly, we collect more specific information about the events, such as the number of time certain special keys like tab and backspace were used, the number of times a user pastes text, a total count of the number of unique keys a user presses, and how many form fields were accessed. This information can help expose users with especially unique behavioral patterns. Finally, we collect information about the delays the user introduces into their work. We determine how long the user spent 'off focus' from the page, the cumulative time they spent before they started typing in a form field, and the cumulative time they spent between keystrokes in a form field. We can use these features to make higher level judgments about user deliberation and attention in tasks. For crowdsourcing markets such as Mechanical Turk, we incorporate the time the mar-

General Information

Raw Log Length	Assignment ID
Refined Log Length	Worker ID
Discretized Log Length	

Timing

Total Task Time	Before Typing Delay
On Focus Time	Within Typing Delay
Recorded Time Disparity	

Action Counts

Total Clicks	Total Keypresses
Total Mouse Movement	Total Pastes
Total Scrolled Pixels	Total Tabs
Total Fields Accessed	Total Backspaces
Total Focus Changes	Count of Unique Characters

Table 1: Aggregate data collected by the prototype

ket reports they spent on the task, and their unique worker identification.

Our implementation exposed several limitations in event logging using Javascript. Foremost, cross-browser compatibility, despite many standardization efforts, remains poor. Some browsers accurately report when the user focus changes from a window, others provide no feedback whatsoever. Some browsers report keystrokes accurately, others provide questionable information about special characters. Likewise, browser extensions that prevent JavaScript from running, such as NoScript, completely inhibit logging. Latency is also an issue, as sending events as they occur to the server proved to be less reliable than a one-shot upload at the end of the task. As implemented presently, there is little to no latency in our logging scheme since the user data is uploaded at the end of the task. In future work we will explore more reliable means of periodic updating. Finally, user logging has privacy implications, as such a script can send sensitive user data without any signals to the user. Our decision to use an opt-in approach addressed both the latency and privacy issues in a way that fit the Mechanical Turk market; however the technique does not require this be the case.

EXPLORATION AND EVALUATION

To evaluate the utility of task fingerprinting, we examined collections of workers performing a series of tasks on Amazon's Mechanical Turk crowdsourcing market. Mechanical Turk is a rich arena for studying tasks, as workers are willing and able to complete jobs ranging from the minute, like identifying parts of speech, to the complex, like comprehending a passage or composing a summary. Further, there are enough workers on Mechanical Turk that one is likely to get a good sample of worker behaviors for a given task [15, 20].

We conducted three different experiments on Mechanical Turk designed to highlight a variety of cognitive tasks that crowdworkers may do and the applications task fingerprinting has in such work. Our first experiment has Turkers identify words that are nouns in a list. This task relies on simple word recognition, and can therefore be performed rapidly and workers can complete multiple tasks in quick succession. We can evaluate the results simply by counting labeling errors. Our second experiment has Turkers generate keyword tags for four different images. Since this task is generative, there is a large spectrum of results that a worker can produce, which we evaluate by both the quantity of tags and the descriptive value of the tags. Our final experiment has Turkers read a passage and answer reading comprehension questions. The evaluation, based on correct answers, provides insight into the worker's understanding of the text. For each experiment we solicit workers on Mechanical Turk, capture their behavior using our prototype, validate that our observations align with their end products, develop a task fingerprint using our observations, and apply the task fingerprint to the evaluation of the task using Weka, a machine learning toolkit [8].

Classification Tasks

We had Mechanical Turk workers perform data labeling, a type of task often used on human computation markets. Workers were presented with a HIT (Human Intelligence Task) that presented them with a list of 40 words and asked them to check boxes for words that were nouns and leave non-nouns unchecked. On average, each HIT had 11 nouns and 29 verbs, adjectives, or adverbs between 4 and 9 characters selected from the Moby and Wordnet databases intersected with an English as a second language dictionary so as to choose easier words. Payment was set at $0.05, somewhat high for a task of its magnitude, in order to encourage cheating as well as unscrupulous behavior. We

Task	Input Type	Submissions	Evaluation Measures
Classification	word ☑	185	Noun identification accuracy
Content Generation	tag	114	Rated quality, 'cheated' label
Comprehension	A ○ B ◉	63	Question answering accuracy

Table 2: Experimental conditions

solicited a total of 5 instances of each of 40 different labeling tasks, totaling 200 requests. Of those 200 requests, 15 were excluded because their browsers did not relay event logs. 21 unique participants generated the remaining 185 points in this task.

We evaluated participants based on the number of 'correct' answers they gave, where a correct answer meant checking a noun and leaving a non-noun unchecked. On average people correctly classified 83% of words (*SD*=14.1), compared to an average of 73% if they had left the form completely blank. Because the participant average is below even what would be the case if they only checked half the nouns and left the rest blank (86%), it is likely that a fair percentage of workers put forward a minimum amount of effort.

Results: Task Time

While task time can be a useful metric for identifying non-compliant workers, as researchers using Mechanical Turk are well aware, task times reported by the system can be dramatically different from the actual time a worker spent on completing a task [15]. Workers may often accept multiple tasks and leave them open while finishing others. Our data enables us to examine how large the discrepancy is between reported and actual work time for tasks. Examining all of the worker logs, which averaged 32.7 refined events, we discover that workers spent on average 1 minute, 13 seconds completing the task and clicked on checkboxes 9.5 times. The time of task completion recorded by Mechanical Turk's system was off by an average of 21.1 seconds from recorded on-task time. As shown in Figure 2, this disparity is unevenly distributed, with a few tasks accepted but not worked on for a long time period. Interestingly, tasks with high reported vs. actual time discrepancies may also tend to be of lower quality. In our later reading comprehension task, higher time disparity was marginally correlated with more errors ($F(1,62) = 3.28$, $p=0.075$) This suggests that although the raw reported time from Mechanical Turk may not be dependable as a measure of effort (as workers may accept and queue tasks), using the disparity between reported and actual time may actually offer some information about the quality of the work.

Results: Accuracy

We used machine learning to predict our quantitative evaluation of the labels each Turker provided using the task fingerprints. After removing non-essential features such as keyboard presses and delays (since there was no typing),

First, we investigated a binary prediction task, using a pass/fail threshold of 80% (where "pass" corresponds to a generous threshold of identifying 3 nouns accurately with no non-nouns checked; 69 of 185 participants fail this milestone and 116 pass). This threshold also is consistent with the 30% cheating ratio found by other crowdsourcing researchers [1, 15]. After identifying the most predictive features, we used Weka to generate decision trees to predict

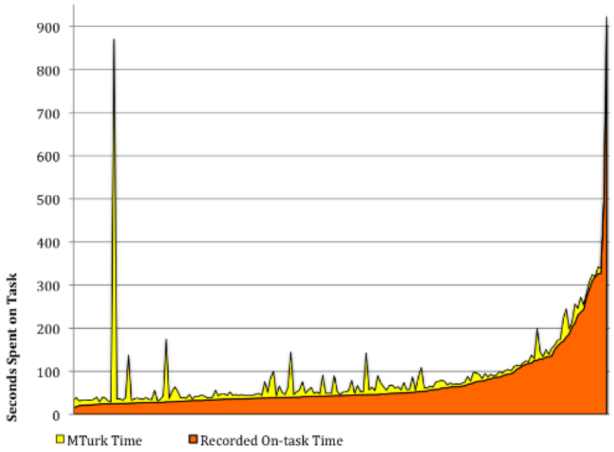

Figure 2: Time disparity between our logged 'on task' time and MTurk's recorded task time (in seconds) on the word classification task.

our pass/fail classification[1]. We limited the number of features used in these trees to maximize generality and avoid overfitting. Our initial tree utilized the number of clicks, checkboxes accessed, and the difference between the Turk recorded time and our event log time. Using 10-fold cross-validation, the model predicted our pass/fail evaluation for the 185 data points with 83.2% accuracy, a kappa of 0.608, and an F-measure of 0.823. This suggests such a model could highlight points of interest for exclusion or human inspection. However, since many of the checkboxes were correct in their default unchecked form, we considered the possibility that the number of fields accessed may be too directly tied to our choice of leaving nouns in the minority. Removing those fields, we generated a decision tree that utilized the total amount a user scrolls and moves the mouse as well as the disparity between recorded task times. This model, using only summary statistics about the user's behavior, classified the points with 78.3% accuracy, a kappa of 0.534, and an F-measure of .784, reinforcing our suggestion that even with limited fingerprint data, a model could highlight questionable points in a large sample of end products.

Beyond classifying workers' products as suspect, we investigated whether we could predict the raw accuracy score of a given worker using only their fingerprint. Using support vector regression, we trained models from the fingerprints and accuracy scores. Under 10-fold cross-validation, our model significantly correlated with the actual accuracies we recorded ($r=0.3289$, $p<0.001$). This suggests the model may be suitable for identifying high quality work in a large sample of completed submissions. By incorporating worker

[1] We initially tested our data with both decision trees and SVMs and found that they both provided similarly high levels of accuracy. We chose decision trees because their small set of features is easy to interpret and describe with respect to user behavior.

Image	Quality Rating	Predicted Rating	Tags
	4.5	4.799	Orange, Storm, Mountain, Night, Clouds, Sky
	3	3.33	Tree, Cloud, Sky
	1	1.02	<no tags, with comment 'he he he!!!!'>

Table 3: Rated quality versus model predicted quality with tags for one of the four images tagged by the worker

identity we further improved the model, boosting the correlation higher ($r=0.8926$, $p<0.001$). Similarly, adding worker identity and predicting a pass/fail score using a decision tree without clicks classifies better than our previous classifier, having an accuracy of 85.4%, a kappa of 0.681, and an F-measure of 0.856. Examining the trees, it is clear that accounting for intra-worker variance has significant benefits, since workers seem to produce similar quality work across multiple iterations of the task.

In summary, we found that task fingerprinting is able to predict the outcomes of simple labeling tasks in Mechanical Turk, even when cheating and low-quality results represent a large proportion of the data. Moreover, limiting the task fingerprint to simple features such as cumulative mouse movement and scrolling is effective. However, human computation tasks are not limited to binary labeling. Next we examine a more complex task with a different set of evaluation criteria.

Content Generation Tasks

To investigate content generation HITs on Mechanical Turk, we had Turkers supply three to five keyword tags for each of four images. We generated three different sets of images based on three themes: art, pets, and landscapes. For each of the themes, we solicited 20 submissions. To gather more variance, we generated a duplicate set of the series of tasks, this time explicitly asking for workers to pretend they were clever cheaters. Their new task was to try to complete the same tagging task with the minimum of effort needed to avoid being caught by an inattentive requester. We requested a similar group of 20 submissions for each image set under this condition. Our examinations of the end products revealed that this 'cheating' group in fact produced many acceptable submissions, suggesting that some of the workers may not have comprehended the nature of the cheating task or that "clever" cheating may actually have been more difficult than doing the task in good faith [15]. As a result, we combined the two datasets into one that represented a broader range of work quality. Of the 120 submissions, we excluded 6 from which we did not receive event logs. The remaining 114 points represent the work of 52 unique participants.

Unlike in the noun identification task, we did not have gold standard images and tags to provide a quantitative evaluation. Instead, we had two raters examine each group of tags with respect to the set of images and judge them on two five point scales. The first scale concerned the *quantity* of work done, where a value of 1 represented clear cheating or no work completed, 3 meant an adequate amount of work according to the HIT directions, and 5 represented exceptional effort. The second scale concerned the *descriptiveness* of completed work, where 1 corresponded to poor quality, specious, or empty tags, 3 represented tags that accurately described the images, and 5 meant exceptionally descriptive tags. The raters rated the 114 points with high interrater reliability (Spearman's $\rho = 0.7541$, 0.7636; $p<0.001$, $p<0.001$ respectively). The two scales are correlated, suggesting they indeed measure an innate *quality* aspect of the task results, as confirmed by their high item reliability (Cronbach's $\alpha = 0.8248$). As a result we averaged the results of the two scales into one rating for general performance, and of the 114 points, the rating for submissions averaged to 3.5 out of 5 ($SD=1.13$). We also had the raters decide by consensus from the submitted tags whether a submission represented cheating. Of our points, 17, or 14.1%, were identified as clear cheats. This proportion is smaller than in our previous experiment likely because the task was more complex and there were a small number of tasks to complete in series, thus making them less attractive to potential cheaters.

We constructed task fingerprints as before from the logs, which averaged 107.9 events. On average, the workers spent 2 minutes, 32 seconds on the task, spending in total an average of 39.7 seconds before they typed a tag in a field, and 30.3 seconds typing their tags. On average they used 20.5 different characters and typed 105.8 keystrokes. The Mechanical Turk system reported times that were on average 27.1 seconds longer than our recorded on-task time.

We investigated whether the fingerprints for image tagging could predict whether a person cheated or not using a logistic decision tree. The resulting tree weighted primarily for the number of unique ASCII characters used and the total time spent on the task. Under 10-fold crossvalidation it achieved 93.0% accuracy, a kappa of 0.655, and an F-measure of 0.930 using only those two attributes. The tree structure suggests that cheaters might use fewer unique keyboard keys (leading to fewer distinct tags) and take less time to complete the task than non-cheaters.

Given this success, we also used support vector regression on the task fingerprints to predict the rated quality of the results. The resulting model significantly predicts quality (correlation with actual ratings: $r = .5874$, $p<0.001$). It sug-

gests that the more fields accessed, more unique characters, fewer total keypresses, more clicks, and more total time spent predict higher scores. In summary, the model can suggest how good tags will be without knowledge of the tags themselves.

We also examined whether our system could predict high quality outcomes, as opposed to just cheaters and low quality output. After filtering the data to only acceptable submissions and higher, we applied support vector regression to the remaining 81 high scoring points. Once again, our model was highly correlated with the actual scores ($r=0.4598$, $p<0.001$). Thus, given only high quality data, we can still predict the quality rating of submitted tags.

Our results suggest that for even qualitative, generative tasks like image tagging, task fingerprints encode information that can help identify cheaters and predict the quality of the tags produced. The predictions relied on low-fidelity statistical information, such as the number of unique keys used and the total time on task. One can imagine quality ratings from the predictions functioning as confidence values for the set of tags produced. From there, tags from many jobs could be ranked cumulatively by order of quality. Yet, while generative tasks represent another area of human computation, there still remain more complex tasks. Does task fingerprinting function in tasks where cognitive processing may be less evident through the keyboard?

Comprehension Tasks

We chose reading comprehension for our evaluation of task fingerprinting in complex cognitive work on Mechanical Turk. We gave workers a passage from an online test preparation booklet in English and had them read the text and then answer questions based on their reading. We ascertained their level of comprehension through 9 questions: 8 multiple-choice, and 1 short response sampled from the same test preparation booklet. Of the 8 multiple-choice questions, we added one question that could be answered easily using only the text from another question without even examining the passage. This allowed us to evaluate whether a Turker was even paying attention to the questions they were answering, or were clicking random answers. We solicited 35 responses for each of two different passages, one corresponding to the physicist Marie Curie, the other expositing on the circumnavigation of the globe. Of the 70 Mechanical Turk results, we received 63 unique event logs from 45 different participants. Only 4 submissions failed the test question, suggesting that by and large the Turkers were at least reading and understanding the questions. On average the workers got 5.32 questions correct out of 9 ($SD=1.97$). Examining the task fingerprints, workers spent an average of 6 minutes, 2.9 seconds on the task, and their task time as reported by Mechanical Turk differs from our recorded on-task time by an average of 37.4 seconds. Refined logs averaged 298.6 events because of scrolling and mouse moment.

As in the previous cases, we used the task fingerprints to predict the performance of workers on the task. In this case, our performance measure is the number of correct answers a worker entered, which approximates their overall learning and comprehension from the passage. Using support vector regression, task fingerprints significantly predicted the comprehension level of Turkers ($r=0.260$, $p=0.0393$). The predictive model depended largely on the time spent on focus, the difference between the recorded HIT time and our event log time, the total mouse and scroll movement, the number of clicks, and the delay between typing characters in the short response. The typing delay might relate to the fact that many successful submissions copy-pasted their answer to the short answer question from the passage. This produces a zero typing delay, which explains the negative relation between delay and number correct. Mouse movement and scrolling might capture the behavior of workers that often refer to the passage when answering questions. Based on these findings, we suggest that task fingerprints indeed hold predictive value for higher cognitive tasks and functions in crowd workers.

Avoiding Manual Labeling

In the previous experiments we utilized fully labeled data. It is likely to be the case that the data used for crowdsourcing is neither perfect nor gold standard. More often than not, it is likely to be unlabeled and hard to evaluate by hand. Given our previous experiments' reliance on predicting labels based on an entire scored dataset, we now investigate three different means to reducing the burden on requesters in actually applying task fingerprinting to crowdsourced tasks.

We conducted test runs of our image tagging and word identification data training on only small randomly selected proportions of the total labeled data points. We posit that if we are able to predict the rest of the dataset with reasonable accuracy, then it is likely that requesters need not label their entire dataset. Rather, they need only label a small subset to provide the necessary training for a task fingerprint predictor. In the case of image tagging, we trained a qualitative performance rating support vector regression model using 5% through 60% of the data, in increments of ten percent, averaging 20 runs that use a different random selection of data points each time. Although the model cannot significantly predict performance using 5% and 10% of the data, for 20% of the data (23 points) and above the model predictions significantly correlate to the actual ratings. There is enough data in the task fingerprints that a small sample and a generalized machine learning model can provide good accuracy. Running a similar prediction for accuracy in our word identification task reinforces this: Once again, from 20% of the data (37 points) onwards the model's predictions correlated with significance to the actual accuracy values. Thus, one way to avoid being overburdened with labeling is to simply label a selection of random points, create a classifier using the task fingerprints, and examine selected results to ensure it is behaving appropriately.

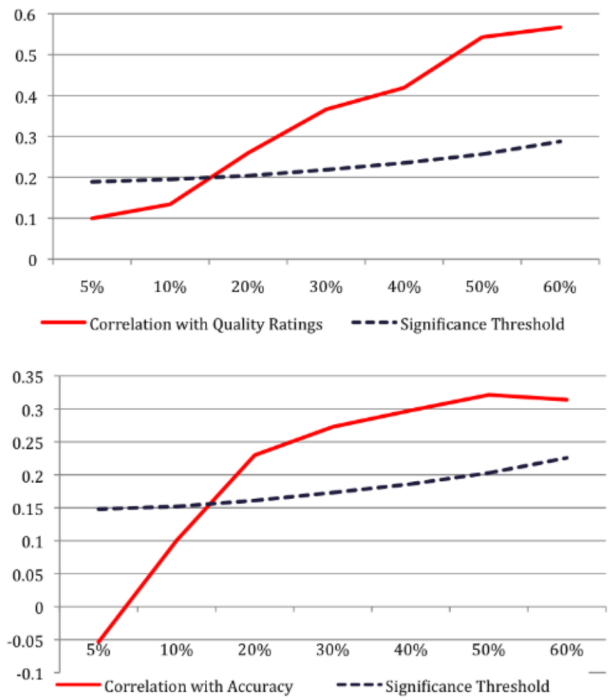

Figure 3: Model prediction correlation with actual ratings as training set size increases for image tagging and word identification

However, labeling data may not be possible for all datasets and tasks. Yet, some tasks are similar to other tasks in Mechanical Turk. For instance, our reading comprehension task involved workers examining a passage and then clicking on multiple choice boxes. Would such a model port over to our word identification task, which asks workers to examine words and click to make a judgment on whether it is a noun? After we normalized all of the task fingerprint values for both reading comprehension and word identification, we trained a support vector regression model on all of the normalized counts of correct answers in the reading comprehension problem. We then applied this model to the entirety of the word identification dataset, predicting its normalized count of correct answers. The model was able to significantly predict correct answers in the new dataset ($r=0.4948$, $p<0.001$). Thus, if one had gold standard data for a congruent task, one may be able to gather task fingerprints for the benchmark job and then apply the model to evaluate a related different task without labels. It is particularly surprising how well the model generalized given the fundamental differences in the nature of the tasks: reading a passage and answering multiple choice questions versus identifying nouns in a word list. Building up a toolbox of archetypal task fingerprints for model training may enable prediction for a variety of tasks and evaluations.

It is possible that even in the absence of any labeled data, a mixed-initiative approach starting with unsupervised clustering could be used to bootstrap the system. By visualizing

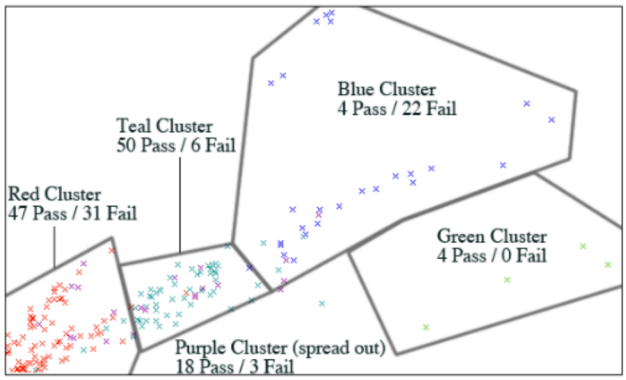

Figure 4: Word identification task fingerprint clusters graphed based on the number of fields each user accessed (x) versus the length of their collapsed event log (y). Notice that the blue and teal clusters discriminate between pass and failure well. The red cluster encapsulates borderline points, while the diffuse purple cluster gathers diffused 'pass' points.

features that differ between clusters (e.g., number of fields clicked on, time on task) employers could identify potential outliers and after investigation label the cheaters; such labels could then be leveraged by the system for the unlabeled data. Prior research using unsupervised clustering techniques for finding differences in problem solving skills between users using event logs demonstrate the potential viability of such an approach [6], suggesting this may be a profitable area for future research.

We conducted a preliminary investigation into the feasibility of unsupervised clustering of task fingerprints. Using the word identification task, we stripped the points of labels and used expectation maximization to identify 5 clusters of fingerprints. Four of the clusters corresponded to high likelihoods of either high or low performance workers, while one cluster was split, warranting manual inspection. Figure 4 shows a visualization of the clusters on two dimensions (fields accessed and collapsed log length); this suggests the potential for a mixed initiative system in which the user could inspect representative cluster samples and outliers, bootstrapping the classification process.

CONCLUSION

This paper introduces and evaluates task fingerprinting, a method for capturing crowdworker behavior and making inferences about their task performance. We demonstrate that by analyzing a worker's procedure in a generalizable way, we can develop event logs and behavioral statistics that can predict outcome measures such as task accuracy, content quality, or comprehension. Furthermore, we find that models trained on one task can predict performance on other related tasks.

We prototyped task fingerprinting by gathering event logs for Amazon Mechanical Turk workers and post-processing the data to extract statistical information about the workers'

processes. We applied this prototype to image tagging, part of speech classification, and passage comprehension experimentally. Training machine learning models on the fingerprints, we were able to accurately predict outcome measures such as qualitative ratings of tags, noun identification accuracy, and passage comprehension, even when training using a small proportion of rated end products.

Future work remains to be done in several areas. Investigating how task fingerprinting might affect 'botting', or automated task completion on markets might be identified using event log pattern detection, for example examining the variance of the workers' behavior (e.g., using string comparison methods like minimum edit distance on refined event logs, or temporal variance measures). This approach could be even more powerful if requesters shared the fingerprints of known bots as they emerged (as antivirus companies do with virus hashes). More varied tasks, including ones where workers might spend significantly different amounts of time and effort on a task might be tested to reinforce the consistency and comparability of fingerprints across workers and tasks. Clustering of task fingerprints not based on statistical data, but rather the conformation of the event log strings using bioinformatics algorithms might also yield useful behavioral information. Moreover, there remains much work to be in visualizing task fingerprints for human inspection and analysis.

Finally, it may be advantageous to combine task fingerprints with other forms of task performance predictors. Our approach has both advantages and drawbacks that are complementary to other approaches to improving crowd worker output, including both task design methods and post-hoc approaches based on gold standards or worker agreement or reputation. Exploring appropriate ways to combine these metrics is an important area for future work.

ACKNOWLEDGMENTS

We would like to thank Jen Mankoff for her valuable feedback and the Carnegie Mellon Social Computing Lab for their support. This work was supported by NSF grants OCI-0943148 and IIS-0968484, and the Center for the Future of Work, Heinz College, Carnegie Mellon University

REFERENCES

1. Bernstein, M.S., Little, G., Miller, R.C., et al. Soylent: a word processor with a crowd inside. *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, ACM (2010), 313–322.
2. Callison-Burch, C. Fast, cheap, and creative: Evaluating translation quality using Amazon's Mechanical Turk. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, Association for Computational Linguistics (2009), 286–295.
3. Chi, E.I., Pirolli, P., and Pitkow, J. The Scent of a Site : A System for Analyzing and Predicting Information Scent , Usage , and Usability of a Web Site. *2*, 1 (2000).
4. Dekel, O. and Shamir, O. Vox populi: Collecting high-quality labels from a crowd. *COLT 2009: Proceedings of the 22nd Annual Conference on Learning Theory*, Citeseer (2009).
5. Downs, J.S., Holbrook, M.B., Sheng, S., and Cranor, L.F. Are your participants gaming the system?: screening mechanical turk workers. *Proceedings of the 28th international conference on Human factors in computing systems*, ACM (2010), 2399–2402.
6. Fern, X., Komireddy, C., Grigoreanu, V., and Burnett, M. Mining problem-solving strategies from HCI data. *ACM Transactions on Computer-Human Interaction 17*, 1 (2010), 1-22.
7. Ghazarian, A. and Noorhosseini, S.M. Automatic detection of users' skill levels using high-frequency user interface events. *User Modeling and User-Adapted Interaction 20*, 2 (2010), 109-146.
8. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I.H. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter 11*, 1 (2009), 10–18.
9. Hilbert, D. and Redmiles, D. Extracting usability information from user interface events. *ACM Computing Surveys (CSUR) 32*, 4 (2000), 384-421.
10. Huang, E., Zhang, H., Parkes, D.C., Gajos, K.Z., and Chen, Y. Toward Automatic Task Design : A Progress Report. *Proceedings of the ACM SIGKDD workshop on human computation*, ACM (2010), 77–85.
11. Hurst, A., Hudson, S., Mankoff, J., and Trewin, S. Automatically detecting pointing performance. *Proceedings of the 13th*, (2008), 11.
12. Ipeirotis, P.G., Provost, F., and Wang, J. Quality management on amazon mechanical turk. *Proceedings of the ACM SIGKDD workshop on human computation*, ACM (2010), 64–67.
13. Ivory, M. and Hearst, M.A. The state of the art in automating usability evaluation of user interfaces. *ACM Computing Surveys (CSUR) 33*, 4 (2001), 470–516.
14. Kim, J., Gunn, D., Schuh, E., and Phillips, B. Tracking real-time user experience (TRUE): a comprehensive instrumentation solution for complex systems. *Proceedings of the twenty-sixth annual SIGCHI conference on Human Factors in Computing Systems*, (2008), 443-451.
15. Kittur, A., Chi, E., and Suh, B. Crowdsourcing user studies with mechanical Turk. *Proceedings of the twenty-sixth annual SIGCHI conference on Human Factors in Computing Systems*, (2008), 1509-1512.
16. Mason, W., Street, W., and Watts, D.J. Financial Incentives and the " Performance of Crowds ."*SIGKDD Explorations 11*, 2 (2009), 100-108.

17. Rauterberg, M. and Aeppli, R. Learning in Man-Machine Systems : the Measurement of Behavioural and Cognitive Complexity. *Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on*, IEEE (1995), 4685–4690.

18. Rogstadius, J., Kostakos, V., Kittur, A., Smus, B., Laredo, J., and Vukovic, M. An Assessment of Intrinsic and Extrinsic Motivation on Task Performance in Crowdsourcing Markets. (2011).

19. Shahaf, D. and Horvitz, E. Generalized task markets for human and machine computation. *Proc. 24th AAAI Conference on Artificial Intelligence*, (2010).

20. Snow, R., O'Connor, B., Jurafsky, D., and Ng, A.Y. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics (2008), 254–263.

21. Stieger, S. and Reips, U.-D. What are participants doing while filling in an online questionnaire: A paradata collection tool and an empirical study. *Computers in Human Behavior 26*, 6 (2010), 1488-1495.

22. Vanderaalst, W., Vandongen, B., Herbst, J., Maruster, L., Schimm, G., and Weijters, a. Workflow mining: A survey of issues and approaches. *Data & Knowledge Engineering 47*, 2 (2003), 237-267.

23. Von Ahn, L. and Dabbish, L. Labeling images with a computer game. *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM (2004), 319–326.